La representación de algoritmos diseñados bajo la técnica "divide y vencerás"

Article · January 2011

DOI: 10.34624/id.v3i3.4529

CITATIONS
2

READS
1,487

3 authors:

Antonio Pérez Carrasco
King Juan Carlos University
33 PUBLICATIONS **220** CITATIONS
SEE PROFILE

J. Ángel Velázquez-Iturbide
King Juan Carlos University
262 PUBLICATIONS **2,306** CITATIONS
SEE PROFILE

Francisco J. Almeida-Martínez
University Isabel I
21 PUBLICATIONS **46** CITATIONS
SEE PROFILE

Tecnologias da Informação em Educação

La representación de algoritmos diseñados bajo la técnica "divide y vencerás"

Antonio Pérez Carrasco
Universidad Rey Juan Carlos, España
antonio.perez.carrasco@urjc.es
Jesús Ángel Velázquez Iturbide
Universidad Rey Juan Carlos, España
angel.velazquez@urjc.es
Francisco Almeida Martínez
Universidad Rey Juan Carlos, España
francisco.almeida@urjc.es

Resumen

Hacemos un recorrido por una extensa bibliografía para catalogar y valorar el repertorio de visualizaciones gráficas que emplean diversos autores para representar algoritmos de la técnica de diseño *"divide y vencerás"*. Además, comparamos las visualizaciones encontradas con las que aporta SRec, aplicación informática desarrollada en el seno de nuestro grupo de investigación orientada a facilitar las tareas de docencia y análisis de algoritmos haciendo uso de visualizaciones animadas de carácter interactivo. Por último, intentamos proponer algunas visualizaciones para representar, de una manera genérica pero eficaz, aquellos algoritmos para los que actualmente ni los libros ni nuestra aplicación dan una representación útil.

Palabras-clave: algoritmos; divide-y-vencerás; libros; visualización.

Indagatio Didactica, vol. 3(3), dezembro 2011 ISSN: 1647-3582

Resumo

Fizemos uma revisão de uma extensa bibliografia para catalogar e avaliar o repertório de visualizações gráficas que adotam diversos autores para representar algoritmos da técnica de desenho "divide e vencerás". Por outro lado, comparámos as visualizações encontradas com as que nos oferece a SRec, uma aplicação informática desenvolvida no seio do nosso grupo de investigação orientada para facilitar as tarefas de docência e análise de algoritmos fazendo uso de visualizações animadas de carácter interativo. Por último, tentamos propor algumas visualizações para representar, de una maneira genérica mas eficaz, aqueles algoritmos para os quais atualmente nem os livros nem a nossa aplicação oferecem uma representação útil.

Palavras-chave: Algoritmos; Divide-e-vencerás; Livros; Visualização.

Abstract

We conducted a review of an extensive bibliography to catalogue and evaluate the repertoire of graphic visualisations that adopt several authors to represent algorithms of the "divide-and-conquer" drawing technique. On the other hand, we compared the visualisations found with those offered by SRec, a computer application developed in our research group oriented towards rendering easy the teaching tasks and the analysis of algorithms making use of animated visualisations of interactive nature. Finally, we try to propose a few visualisations to represent, in a generic but effective manner, those algorithms for which at present neither books, nor our application, offer a useful representation.

Key-words: Algorithms; Divide-and-conquer; Books; Visualisation.

1. Introducción

La amplia bibliografía existente centrada en la divulgación de la algoritmia se ha apoyado desde siempre en representaciones gráficas para complementar las explicaciones textuales y aportar ejemplos que ilustren conceptos y ejemplos específicos de los algoritmos que se desarrollan en las obras.

Sin embargo, son muchos los casos en que tales representaciones gráficas se ven superadas por las visualizaciones que son capaces de generar diferentes aplicaciones informáticas desarrolladas en las últimas tres décadas. Estas aplicaciones no sólo aportan visualizaciones de carácter estático, sino que añaden la capacidad de animación y la posibilidad de interacción para ampliar las posibilidades de uso durante la tarea del aprendizaje, ayudando activamente en procesos de comprensión y análisis.

En esta contribución se propone como objetivo el análisis de las representaciones gráficas contenidas en un amplio catálogo bibliográfico, intentando distinguir sus características, puntos fuertes y carencias para compararlas con la aportación que realiza SRec (Velázquez, 2008), aplicación desarrollada por los autores para la animación de programas mediante representaciones genéricas, centrada en algoritmos recursivos y diseñados bajo la técnica "divide y vencerás" (Velázquez, 2009), pudiendo hacer así una evaluación cualitativa de la herramienta.

En el apartado 2 se listan los diferentes problemas, catalogados en distintas categorías para poder extraer generalidades sobre similitudes y carencias de cada tipo. En el apartado 3 se exponen algunas propuestas para algunos problemas o de generalización de ciertas representaciones de operaciones. En el apartado 4 se expresan las conclusiones y quedan indicados algunos posibles trabajos futuros.

2. Los Algoritmos Diseñados bajo la Técnica "Divide y Vencerás"

El catálogo existente de algoritmos diseñados bajo la técnica "divide y vencerás" es muy amplio, si bien se pueden establecer diversas clasificaciones atendiendo a múltiples parámetros que permiten agruparlos en función de ciertas características de los propios problemas. Se repasa a continuación el procedimiento llevado a cabo y los resultados obtenidos, en forma de clasificación de problemas, que son repasados para aportar si cuentan o no con representación gráfica en la bibliografía consultada.

2.1. Recolección de Algoritmos

Partiendo de la hipótesis de que las representaciones empleadas en los libros de referencia son las más extendidas, aceptadas y útiles en el contexto docente, se decidió iniciar una exploración de la bibliografía accesible a través de los recursos que proporciona la biblioteca universitaria seleccionando aquellas obras centradas en el diseño de algoritmos y que cuentan con capítulos centrados en la técnica de diseño "divide y vencerás".

Lo que se buscaba, concretamente, era cualquier tipo de representación gráfica empleada para ilustrar, en un sentido general o con un ejemplo concreto, algún concepto o algoritmo, ya fuese de manera parcial o total. Representaciones como diagramas, secuencias cronológicas, elementos esquemáticos o representaciones especiales, encajan en la búsqueda realizada.

Esta búsqueda se realizó en 13 obras bibliográficas (Di Batista, 1999) (Johnsonbaugh, 2004) (Sahni, 2005) (Cormen, 2005) (Alsuwaiyel, 1999) (Brassad, 1997) (Kleinberg, 2006) (Giusti, 2001) (Brassad, 1996) (Lee, 2007) (Martí, 2004) (Gonzalo, 1997) (Allen, 1995), anotando qué problemas tenían alguna representación, qué tipo de representación era, y qué características singulares tenía.

De esta forma se logró obtener un catálogo de 20 problemas diferentes cuyo punto en común era la de estar diseñados bajo la técnica "divide y vencerás". Todos los problemas encontrados fueron anotados, incluso si no contaban con ninguna representación gráfica. Esto permitió comenzar a distinguir tipos de problemas que no solían admitir representaciones gráficas de otros que ofrecían un catálogo variado.

2.2. Agrupación de problemas

La agrupación de algoritmos permite analizar más cómodamente las carencias que se puedan detectar en ciertos grupos o los factores que permiten generar más fácilmente visualizaciones genéricas expresivas para ciertos algoritmos. Ésta se ha llevado a cabo en base a criterios estructurales de los problemas, lo que ha permitido establecer la siguiente clasificación:

1) Algoritmos que resuelven problemas de carácter matemático, con o sin estructuras de datos (por ejemplo, multiplicación de matrices o factorial). Son algoritmos poco dados inicialmente a la representación gráfica pero que en algunos casos sí logran encontrar visualizaciones que ayuden a comprender su funcionamiento.

2) Algoritmos que hacen uso de estructuras de datos (vectores, matrices) devolviendo un valor de retorno (por ejemplo, algoritmo de búsqueda, maneja un vector y devuelve un valor entero).

3) Algoritmos que hacen uso de estructuras de datos (vectores, matrices), realizando modificaciones sobre tales estructuras, siendo ése el propio resultado del algoritmo (por ejemplo, algoritmo de ordenación, donde el resultado queda en la propia estructura manejada), no devuelven ningún valor específico.

4) Algoritmos con una componente geométrica, manejando puntos en el espacio o incluso polígonos y haciendo uso de reglas matemáticas para cálculos diversos sobre tales elementos. Existe un quinto grupo que aglutina los restantes algoritmos.

Ésta es una clasificación similar a la que propone Stern (2002), basada en el tratamiento de datos que se hace de las estructuras de datos suministradas a la entrada, lo que permite definir tres categorías:

1) recorrido por una estructura (por ejemplo, algoritmo de búsqueda).

2) manipulación de elementos de una estructura (por ejemplo, algoritmo de ordenación).

3) construcción de datos en una estructura (por ejemplo, algoritmo de inserción).

Así, las categorías 2 y 3 podrían corresponderse en gran medida con la categoría 3 de la clasificación propuesta, mientras que la categoría 1 de la catalogación de Stern se asemeja a la categoría 2 de nuestra propuesta.

A continuación se repasan las cuatro categorías de la clasificación propuesta, explorando las visualizaciones disponibles en libros y las que aporta SRec para cada problema. Se podrá observar que los problemas de las distintas categorías cuentan con representaciones muy similares en la bibliografía.

2.3. Algoritmos Matemáticos

Los algoritmos matemáticos dan una respuesta a problemas que buscan solucionar cuestiones teóricas y de carácter abstracto que provienen desde el campo de la ciencia de las matemáticas. Se reúnen en esta categoría problemas como:

1) la multiplicación de enteros de gran tamaño;

2) la multiplicación de matrices con algoritmos avanzados;

3) la exponenciación;

4) el cálculo de factoriales;

5) el cálculo de la serie de números de Fibonacci;

6) otros más complejos como la convolución de funciones y la transformada de Fourier.

Todos estos problemas admiten la búsqueda de una solución desde el enfoque de la técnica "*divide y vencerás*" pero una representación gráfica útil de algunos de ellos no ha sido encontrada aún, por lo que la explicación en libros y clases de los mismos se basa en el desarrollo textual y las expresiones matemáticas. Es el caso de la multiplicación de números grandes (Brassard, 1996) (Brassard, 1997) (Kleinberg, 2006) (Alsuwaiyel, 1999), la multiplicación de matrices en todas sus variantes (Brassard, 1996) (Brasard, 1997) (Sahni, 2005) (Johnsonbaugh, 2004) (Alsuwaiyel, 1999), del cálculo de convoluciones y del de transformadas de Fourier (Kleinberg, 2006) (Lee, 2007). Para ellos, tampoco SRec ofrece una visualización satisfactoria.

Algunos de ellos hacen uso de sencillos diagramas que intentan situar al lector o explicar propiedades o algoritmos de carácter meramente matemático, como por ejemplo, la representación estrictamente matemática de la multiplicación de matrices (Shani, 2005) (Johnsonbaugh, 2004), o la representación de la multiplicación de números, escribiendo uno debajo del otro y dejando huecos para el algoritmo iterativo clásico que se aplica en el cálculo manual (Kleinberg, 2006) y haciendo sobre ellos una primera división en dos mitades que sirva para introducir el algoritmo (Brassard, 1996).

Problemas matemáticos que sí admiten una representación gráfica que ayude a su comprensión y análisis son el cálculo del factorial, la exponenciación y el cálculo de la serie de los números de Fibonacci. De ellos sólo el cálculo del factorial cuenta con representación gráfica en los libros consultados (De Giusti, 2001) para mostrar el comportamiento de la pila del programa, mientras que para la exponenciación (Brassard, 1996) (Brassard, 1997) y el cálculo de la serie de los números de Fibonacci (Brassard, 1996) los autores sólo se apoyan en explicaciones textuales y expresiones matemáticas. Sin embargo, sí que admiten una fácil representación gráfica que es proporcionada por SRec tal y como aparece en la Figura 1.

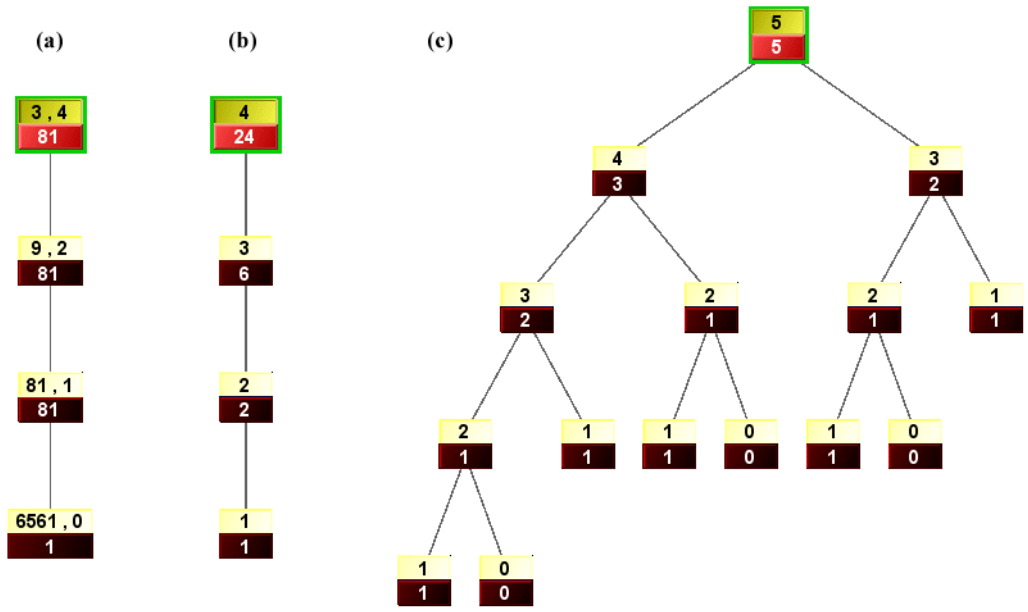

Figura 1 – (a) Representación del árbol para el algoritmo de exponenciación (base 3, exponente 4); (b) Representación del árbol para el algoritmo del cálculo del factorial de 4; (c) Representación del árbol para el algoritmo del cálculo del quinto número de la sucesión de Fibonacci.

Para el problema de multiplicación de números enteros de gran tamaño se da una propuesta en la sección 3. Por su parte, el problema de multiplicación de matrices no puede admitir una representación genérica fácil de diseñar ya que existen numerosos algoritmos que cumplen esta labor ofreciendo unos procedimientos diferentes cada uno de ellos.

2.4. Algoritmos con Estructuras con Valor de Retorno

Estos algoritmos manejan en todos los casos una estructura simple de datos, como un vector o matriz y retornan un valor como resultado de su ejecución que puede ser de cualquier tipo, pudiendo ser a su vez otro vector o matriz. En este grupo se encuentran los problemas de:

1) búsqueda binaria

2) cálculo del máximo de un vector

3) cálculo del mínimo y máximo de un vector;

4) selección (búsqueda del k-ésimo valor más pequeño);

5) cálculo de la mediana del vector resultante de mezclar dos vectores ordenados;

6) determinación de cuál es el elemento mayoritario de un vector, si éste existe.

El problema de la búsqueda binaria cuenta en la bibliografía explorada con dos representaciones principales: una representación en forma de árbol binario balanceado con las posiciones del vector (Cormen, 2005), y una representación que describe los pasos que dan los índices que delimitan la porción del vector que maneja el algoritmo en cada llamada recursiva (Brassard, 1997). La segunda es especialmente interesante para poder comprender cómo el algoritmo va descartando partes del vector basándose en los valores límite de la porción que maneja y en la premisa de que los valores están ordenados.

SRec proporciona visualizaciones muy adecuadas para este problema. A través del árbol de activación SRec ilustra los valores que van tomando los índices, ayudando a ver el valor de retorno justo en el momento en que es encontrado (Figura 2 (a)), mientras que a través de su vista cronológica la selección de las distintas partes del vector se representa de manera totalmente explícita, complementando la vista del árbol de activación. Además, la vista de estructura (Figura 2 (b)) refleja muy esquemáticamente cuántas llamadas recursivas han tenido lugar y sobre qué partes del vector han trabajado.

Figura 2 – (a) Representación del árbol para el algoritmo de búsqueda binaria sobre el vector {27,29,34,36,38,41,48,50}; (b) Representación de la vista de estructura para el algoritmo de la búsqueda binaria sobre el mismo vector.

La representación de los problemas de búsqueda del valor máximo y de la búsqueda simultánea del valor máximo y mínimo de un vector usan representaciones muy similares. La más intuitiva es la que contiene el árbol que representa las reiteradas divisiones del vector en un orden descendente y la transmisión de los valores resultantes parciales en orden ascendente (Lee, 2007) (Sahni, 2005). La utilización de flechas puede ayudar a entender el sentido en que se desplaza cada elemento de la representación.

SRec también hace uso del árbol de activación para este caso, pues deja ver paso a paso gracias a su animación cómo se van realizando las divisiones sobre el vector y cómo se van obteniendo los diferentes valores. Se ofrece un ejemplo en la Figura 3.

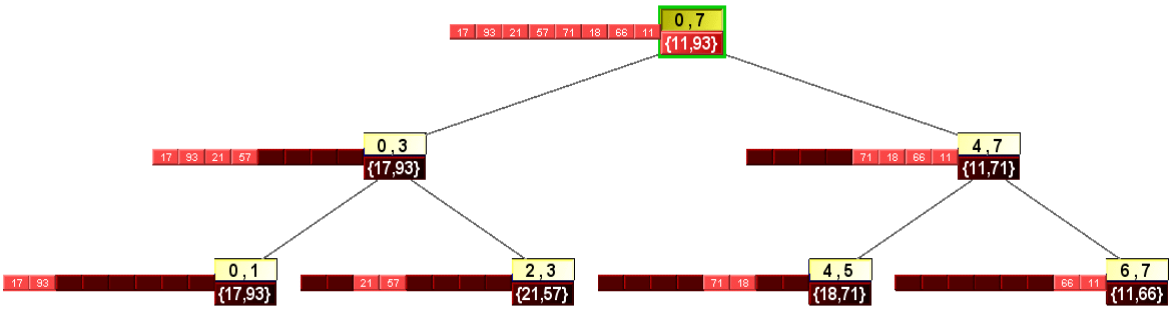

Figura 3 – Representación del árbol para el algoritmo de la búsqueda de los valores máximo y mínimo del vector {17,93,21,57,71,18,66,11}.

El problema de selección es capaz de devolver el valor k-ésimo más pequeño de un vector no ordenado basándose en cálculos de pivote para la división del vector. Este algoritmo cuenta con una representación en libros que refleja en orden secuencial qué partes del algoritmo se van manejando y cuáles se van desechando (Brassard, 1997). Además, se separan las posiciones del vector que contienen la posición del pivote y aquellas que dejan de ser de interés para la búsqueda quedan ocultadas.

SRec, por su parte, ofrece dos vistas idóneas para hacer el seguimiento de la ejecución del algoritmo. En el árbol de activación (Figura 4) se muestran las llamadas sobre la función principal, de carácter recursivo, y sobre la función auxiliar que calcula y ubica el pivote, por lo que queda reflejado muy fielmente el proceso por el cual se va ordenando parcialmente el vector para poder determinar el valor solicitado. La vista de estructura permite complementar la información suministrada por el árbol centrándose en el vector y mostrando de manera gráfica la ubicación del algoritmo en una determinada parte del vector.

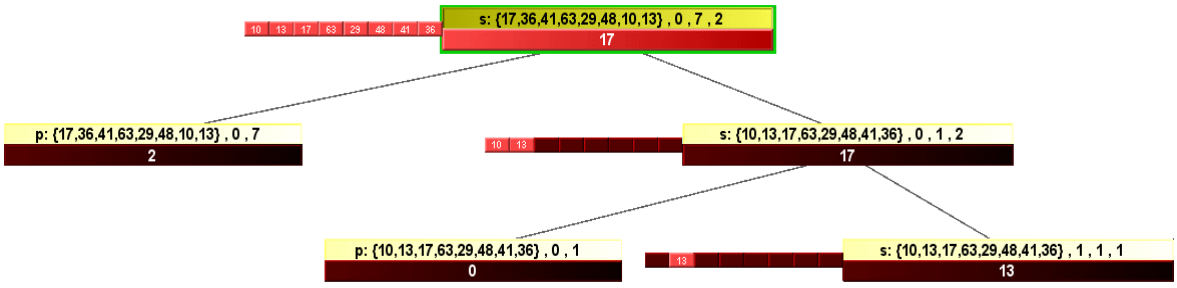

Figura 4 – Representación del árbol para el algoritmo de selección del tercer elemento más pequeño (valor 2 representado como cuarto parámetro) del vector {17,36,41,63,29,48,10,13}. La función "s" es la función principal, de carácter recursivo, mientras que la función "p" es la función auxiliar para el cálculo del pivote.

El problema del cálculo de la mediana del vector resultante de mezclar dos vectores ordenados tampoco cuenta con una representación gráfica que especifique cómo se procede durante el algoritmo para encontrar el valor resultante. SRec ofrece la vista del árbol de activación para poder realizar el seguimiento del algoritmo. En ella se puede ver qué partes de los dos vectores se van utilizando y descartando secuencialmente hasta obtener el valor que tendría la mediana. Para este algoritmo sería útil que en SRec se implementasen algunas de las representaciones que se proponen en la Figura 8.

Por último, el algoritmo que comprueba la existencia de un valor mayoritario en un vector no cuenta con ninguna representación gráfica en la bibliografía manejada. SRec proporciona, una vez más, el árbol de activación que deja ver cómo se comporta el algoritmo y la función auxiliar de cálculo de candidato.

2.5. Algoritmos con Estructuras sin Valor de Retorno

Este grupo alberga diversos algoritmos que manipulan una estructura simple de datos, como un vector o matriz, pero sin aportar un valor de retorno, pues su ejecución se centra en la modificación del contenido de la citada estructura, cuyo estado final puede ser entendido como valor de retorno. Quedan agrupados aquí algoritmos como:

1) ordenación de vectores Mergesort.

2) ordenación de vectores Quicksort.

3) coloreado del tablero defectuoso.

4) intercambio de partes en un vector.

El algoritmo Mergesort concentra la mayor parte de sus operaciones en la tarea de combinación de resultados, no en vano, se dedica a mezclar los resultados parciales obtenidos previamente en las llamadas recursivas. Este proceso de combinación no suele representarse en la bibliografía por ser muy fácil de comprender conceptualmente. Para el algoritmo general sí suele hacerse uso de un árbol de activación que refleje el estado inicial y el estado final en cada subllamada (Alsuwaiyel, 1999) así como una sucesión de imágenes que van mostrando cronológicamente cómo va variando el contenido del vector a medida que es ordenado (Brassard, 1997) (Sahni, 2005) (Johnsonbaugh, 2004).

Con SRec se ofrecen vistas muy similares, con la ventaja de ofrecer la posibilidad de visualizarlo de manera animada e interactiva para cada ejemplo que desee probar el usuario. Las vistas del árbol de activación y de la sucesión cronológica satisfacen ampliamente las necesidades de profesores y alumnos, pues permiten ver de manera intuitiva cómo se obtienen las distintas ordenaciones parciales realizadas por el algoritmo. Aparece un ejemplo de la vista cronológica en la Figura 5 (a).

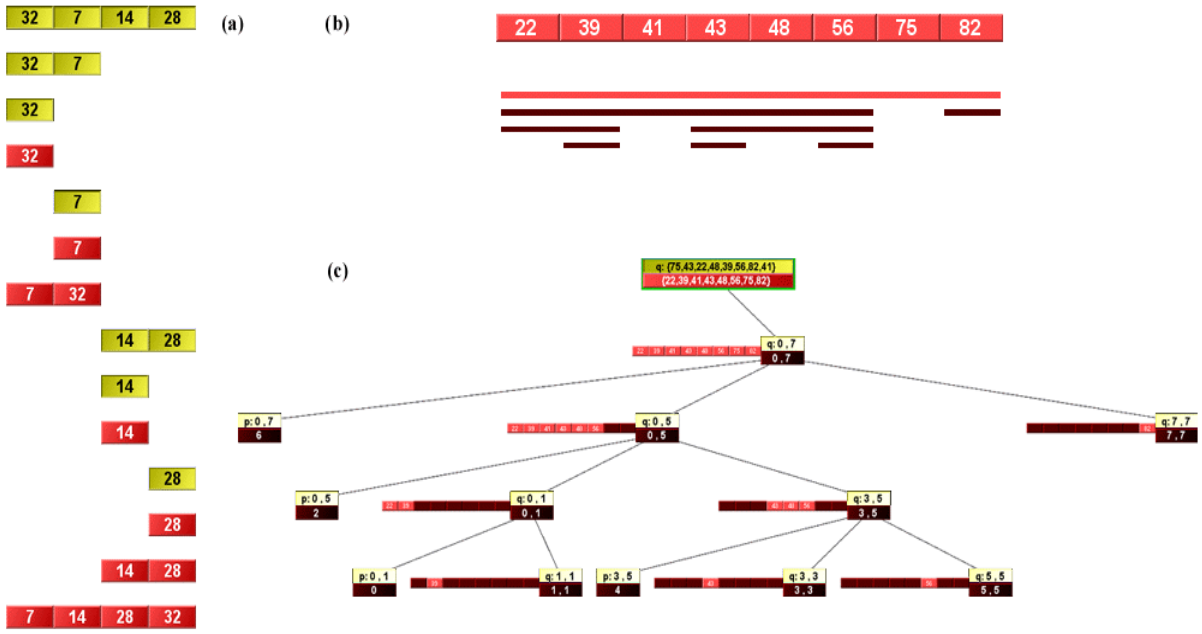

Figura 5 – (a) Representación de la vista cronológica para el algoritmo Mergesort aplicado sobre el vector de cuatro posiciones {32,7,14,28}; (b) Representación de la vista de estructura para el algoritmo Quicksort sobre el vector {22,39,41,43,48,56,75,82}; (c) Representación de la vista del árbol de activación para el algoritmo Quicksort sobre el vector {75,43,22,48,39,56,82,41} donde también se ven las llamadas a la función de cálculo de pivote.

El algoritmo Quicksort, por su parte, realiza sus principales esfuerzos en las tareas de división del vector, calculando la posición del pivote y dejándolo situado en su posición final. Al ser de cierta complejidad conceptual, la bibliografía existente suele centrarse en desarrollar la representación de esta parte paso a paso para un completo entendimiento mediante la representación de secuencias que reflejan cómo se van comparando e intercambiando valores y cómo se sitúa el pivote en su posición final haciendo uso de flechas para representar a los índices y de distintas convenciones (coloreado, doble separación) para remarcar unas partes del vector frente a otras (Brassard, 1997) (Cormen, 2005).

SRec, por su parte, ofrece dos vistas interesantes: el árbol de activación y la vista de estructura. La primera de ellas permite ver paso a paso cómo se va dividiendo el vector, quedando explícitamente marcada la labor del proceso de cálculo del pivote, si bien éste no es mostrado paso a paso como en los libros. Se puede apreciar en la Figura 5 (c). La vista de estructura, gracias a su representación esquemática de la jerarquía de llamadas, da una visión muy simple pero efectiva de qué divisiones se han realizado, dónde han ido quedando ubicados los pivotes y de cuántas llamadas recursivas han sido realizadas y sobre qué zonas del vector. Se muestra esta vista en la Figura 5 (b).

Al igual que ocurre con el algoritmo Quicksort, el problema de colorear un tablero defectuoso con figuras en forma de L concentra en la tarea de división la mayor parte de sus operaciones. Será en ese momento donde se localice el defecto y se coloreen convenientemente los restantes cuadrantes para poder dividir la matriz que se maneja y obtener subproblemas equivalentes al problema original, aunque de menor tamaño. La bibliografía consultada coincide en dibujar la matriz con un elemento en forma de L (Sahni, 2005) o con varios (Johnsonbaugh, 2004) para remarcar el proceso de rellenado.

SRec permite ofrecer mediante sus vistas cronológica y de estructura dos representaciones muy precisas que permiten visualizar con gran detalle los procesos de división y coloreado, resultando muy fácil el seguimiento del algoritmo. Se ofrece una representación de la vista de estructura en la Figura 6 (a).

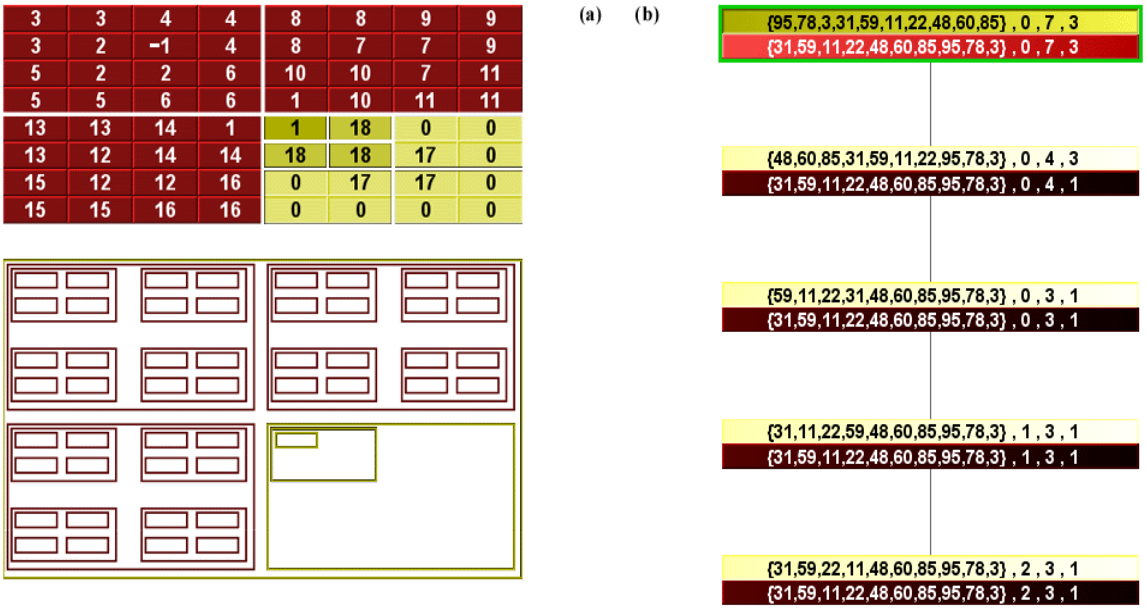

Figura 6 – (a) Representación de la vista de estructura para el algoritmo del tablero defectuoso en un estado intermedio de la ejecución con la posición inicial marcada (valor "-1") en la posición f=1, c=2; (b) Representación de la vista del árbol de activación para el algoritmo de desplazamiento de los valores de aplicado sobre el vector {95,78,3,31,59,11,22,48,60,85}, tomando como objetivo desplazar tres lugares cada posición hacia la izquierda.

Por último, el algoritmo de desplazamiento de los contenidos de un vector es sencillo de comprender y su concepto suele ser fácil de representar, reflejando mediante alguna separación o diferenciación las partes que se van a desplazar (Brassard, 1996). Además, se pueden hacer uso de flechas para representar a los índices que se manejan con el fin de recorrer el vector para ir moviendo los elementos.

SRec, en función de la semántica de los parámetros puede representar mejor o peor a través de las vistas de estructura y cronológica los distintos estados por los que va pasando el vector, pero en todos los casos la vista del árbol permite hacer un seguimiento satisfactorio de las transiciones que se producen en la ordenación de los datos, tal y como se puede ver en la Figura 6 (b).

2.6. Algoritmos Geométricos

Esta categoría reúne algoritmos cuyos elementos tienen un carácter geométrico, como por ejemplo, planos, puntos o líneas. Algunos de los algoritmos recogidos en esta categoría son:

1) cálculo del par de puntos más cercanos entre sí dado un conjunto de puntos en el plano

2) cálculo del número de puntos que domina cada punto del plano dado

3) cálculo de polígonos convexos

4) cálculo de diagramas de Voronoi.

En general, estos algoritmos son representados muy fácilmente en los libros (Alsuwaiyel, 1999) (Cormen, 2005) (Johnsonbaugh, 2004) (Kleinberg, 2006) (Allen, 1995) (Lee, 2007) (Sahni, 2005) haciendo uso de ejemplos donde se sitúan puntos u otros elementos en un eje de coordenadas de dos dimensiones. A estas representaciones se añaden zonas coloreadas, remarcadas, separaciones o bien flechas que sirven para indicar cálculos, relaciones, divisiones del problema, etc.

Para estos problemas, en los que predominan las representaciones bibliográficas basadas en el dominio, las representaciones de SRec resultan muy poco expresivas debido a su carácter genérico y, en general, no aportan ningún valor para ayudar en el análisis o comprensión del algoritmo que se emplea para resolver este tipo de problemas.

2.7. Otros Algoritmos

Aquí se recoge un único algoritmo encontrado en la bibliografía utilizada que no entra en ninguna de las categorías anteriores.

Éste es el problema del cálculo del calendario de un campeonato. Éste intenta determinar el repertorio de encuentros que deben disputarse los contrincantes de un campeonato. El algoritmo es representado con una tabla de manera habitual en la bibliografía consultada (Brassard, 1997). SRec, por su parte, sólo es capaz de ofrecer el árbol de activación como vista para este algoritmo, ya que sus vistas específicas de estructura y carácter cronológico se encuentran preparadas para mostrar exclusivamente matrices cuadradas.

2.8. Resumen de las representaciones usadas por cada categoría de problemas

Se ofrece en la Tabla 1 qué representaciones son utilizadas para representar cada uno de los problemas mencionados, los cuales aparecen agrupados en sus respectivas categorías.

Problemas \ Representaciones	Secuencia esquemática de pasos	Árboles para la expresión de recursión	Diagrama / esquema	Espacial (planos, volúmenes)	Pila de ejecución	Tabular	Textual (ausencia de representación gráfica)
Multiplicación enteros grandes			X				X
Multiplicación de matrices			X				X
Exponenciación							X
Factorial					X		
Serie números de Fibonacci							X
Convolución y Transformada Fourier							X
Búsqueda binaria	X						X
Máximo de un vector		X					
Máximo y mínimo de un vector		X					
Selección			X				X
Mediana de un vector (selección)	X						X
Mediana de vector mezcla de dos vectores							
Elemento mayoritario de vector							X
Mergesort	X	X					X
Quicksort	X	X	X				X
Coloreado tablero defectuoso			X				
Intercambio de partes en un vector		X	X				
Par de puntos más cercanos en plano				X			
Puntos dominados en un plano				X			
Polígonos convexos / Diagramas Voronoi				X			
Campeonato						X	

Tabla 1 – Relación de problemas y representaciones gráficas disponibles de los mismos en la bibliografía consultada.

SRec, por su parte, ofrece varias vistas que se ajustan cada una de ellas en mayor grado a algunos tipos de problemas.

Se refleja en la Tabla 2 para cada problema qué vistas dan un buen resultado (expresado con "X" en la tabla), qué vistas sirven como complemento pero por sí mismas dan un resultado incompleto ("x") y qué vistas pueden usarse para hacer un seguimiento de la ejecución de los algoritmos pero sin ofrecer una contribución de gran interés ilustrativo ("~").

Problemas \ Vistas de SRec	**Árbol**	**Árbol con muestreo de la estructura en nodos**	**Pila**	**Cronológica**	**De estructura**	**Imposible visualización satisfactoria**
Multiplicación enteros grandes	~					
Multiplicación de matrices	~					
Exponenciación	X		X			
Factorial	X		X			
Serie números de Fibonacci	X		X			
Convolución y Transformada Fourier	~					
Búsqueda binaria		X		X	x	
Máximo de un vector		X		X	x	
Máximo y mínimo de un vector		X		X	x	
Selección		X		X	X	
Mediana de un vector (selección)		X		X	X	
Mediana de vector mezcla de dos vectores	~					
Elemento mayoritario de vector		X		X	x	
Mergesort		x		X	X	
Quicksort		x		X	X	
Coloreado tablero defectuoso				X	X	
Intercambio de partes en un vector	~					
Par de puntos más cercanos en plano						X
Puntos dominados en un plano						X
Polígonos convexos / Diagramas Voronoi						X
Campeonato						X

Tabla 2 – Relación de problemas y visualizaciones de SRec con un resultado educativo satisfactorio.

3. Propuestas de representaciones

Tal y como se ha explicado, hay ciertos problemas para los que no se cuenta con una representación gráfica satisfactoria, ni en los libros ni mediante la utilización de SRec.

Vistas las carencias existentes en la representación de algoritmos, se aportan algunas representaciones ideadas para tales algoritmos que intentan ser a su vez lo más genéricas posible para permitir su expansión con otros algoritmos. Estas representaciones están enfocadas a ser integradas en un sistema interactivo de visualización de programas como SRec.

Uno de estos problemas es el cálculo de la multiplicación de números grandes, mencionado en el apartado 2.3. Para este problema se aporta una propuesta en forma de árbol de activación ampliado (Figura 7) que intenta ser genérica y, por tanto, aplicable a más algoritmos mediante el encuentro de una fórmula que permita la representación de operaciones aritméticas basada en ciertas convenciones aplicables en múltiples casos.

Lo que intenta expresar la propuesta que se muestra en la Figura 7 (a) para este algoritmo es la división de los cálculos en tres partes, cada una de ellas multiplicada por la base óptima de exponenciación elevada a 2*s, s y 0 (siendo s el número de cifras que tiene cada parte de los números tras su división en dos mitades). La propuesta de la Figura 7 (b) expresa un algoritmo existente mejorado, de menor complejidad, que también suele ser enseñado en las aulas, en donde se realizan cálculos previos auxiliares sobre tres variables y que posteriormente son utilizadas para ahorrar en el número de multiplicaciones realizadas, que pasan de ser cuatro a tres.

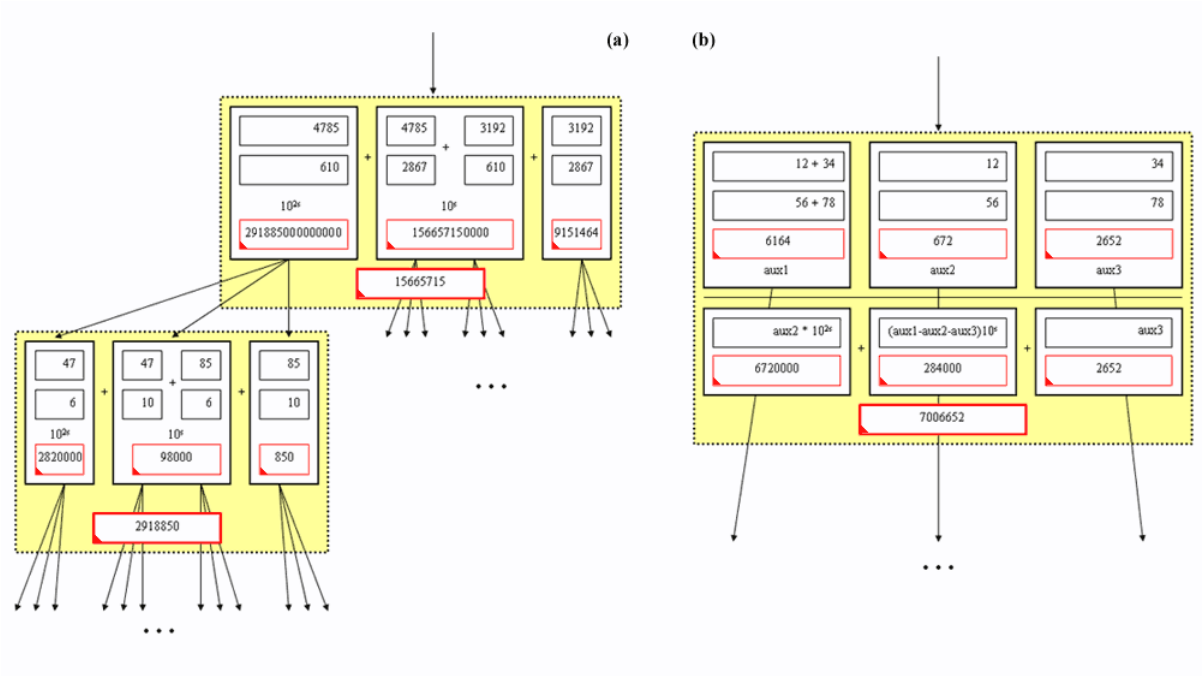

Figura 7 – (a) Representación de la propuesta para el algoritmo de multiplicación de números grandes en una primera variante no optimizada (b) Representación de la propuesta para el algoritmo de multiplicación de números grandes en una segunda variante optimizada del algoritmo.

En ambos casos los árboles se producen por las llamadas recursivas que se lanzan en ambas variantes del algoritmo. En la Figura 7 se muestra el ejemplo para los números 47.853.192 y 6.102.867 (a) y 1.234 y 5.678 (b).

Las flechas inferiores apuntan a nuevas llamadas recursivas, representadas por las celdas amarillas y divididas en varias partes en función de los cálculos que se realizan. Las multiplicación se expresan mediante verticalidad, quedando representado cada operando por una caja, mientras que las sumas se representan de manera horizontal con el operador escrito de manera explícita. Los resultados se expresan en la parte inferior, justo debajo de los operandos, con el borde de la caja en color rojo y un ángulo remarcado.

Esta representación podría extenderse a algoritmos similares en los que las operaciones matemáticas se mantienen presentes de manera importante, estableciendo así un convenio de representación de las diferentes operaciones aritméticas básicas existentes.

Por otro lado, se proponen representaciones genéricas para algoritmos que manejan estructuras tales como vectores y matrices. Estas representaciones, basadas en el árbol de activación clásico, permiten mostrar el contenido de una o varias estructuras de datos, tanto como parámetros de entrada como valores de salida. Cada representación delimita con colores qué partes se están manejando en cada subllamada recursiva, esta delimitación se realiza mediante parámetros, que pueden estar representados en la vista o no.

Pueden ser utilizadas en algoritmos tales como la multiplicación de matrices (sin entrar en detalles específicos de la implementación, pero sí dejando ver los resultados parciales que se van obteniendo) las que aparecen en la Figura 8 (c). No obstante, la multiplicación es sólo una de las múltiples operaciones posibles que se pueden realizar tomando como origen dos matrices, esta representación podría ajustarse a algoritmos que hagan uso de cualquiera de tales operaciones.

También se aportan en la Figura 8 algunas posibles representaciones para algoritmos que manejan dos vectores y que devuelven un valor de retorno, como es el caso del cálculo de la mediana del vector resultante de mezclar dos vectores ordenados. Para este tipo de algoritmos sería totalmente indicada la representación de la Figura 8 (b), teniendo su homóloga con matrices en la Figura 8 (d).

La representación (a) de la Figura 8 podría ser aplicable, por ejemplo, a un algoritmo como el de mezcla de dos vectores que utiliza el algoritmo Mergesort, al emplear dos vectores como valores de entrada y devolver uno nuevo de salida.

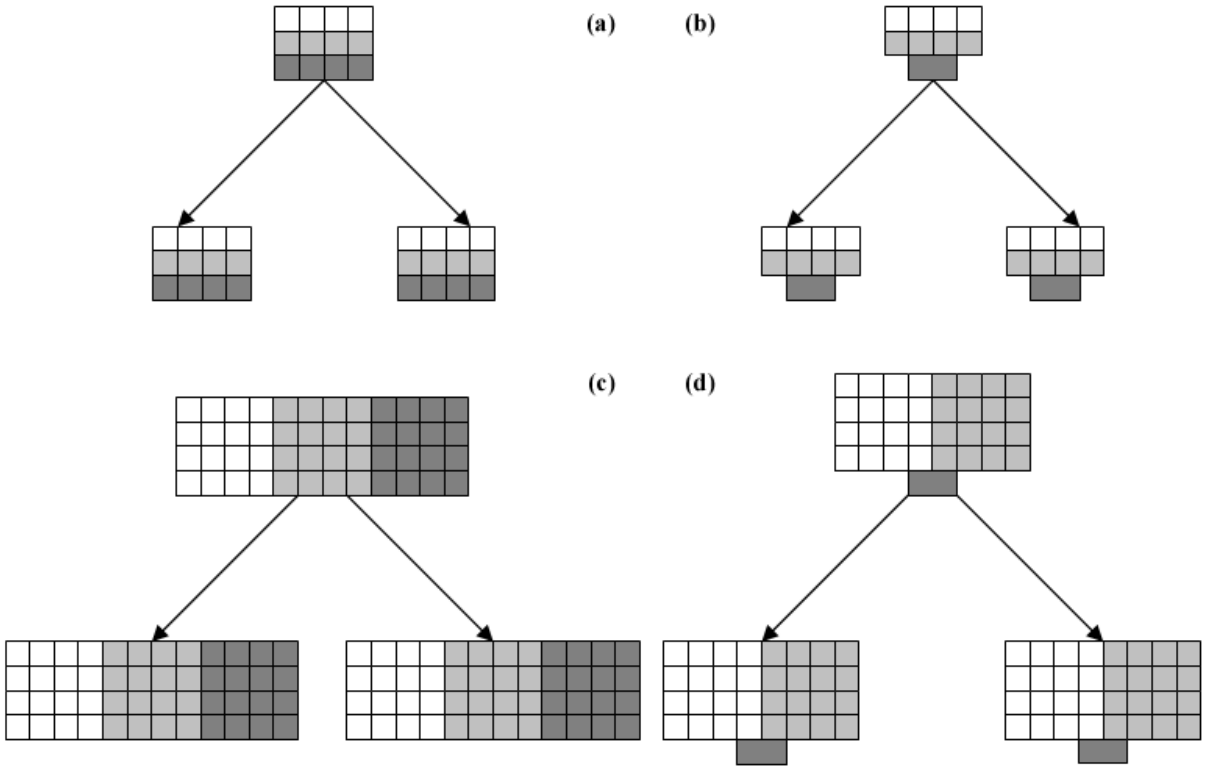

Figura 8 – En todas las representaciones, las posiciones blancas y de color gris claro representan elementos de entrada, mientras que las posiciones de color gris oscuro representan valores de salida; (a) Representación de la propuesta para algoritmos que admiten dos vectores a su entrada y devuelven uno nuevo a su salida (b) Representación de la propuesta para algoritmos que admiten dos vectores a su entrada y devuelven un valor simple a su salida; (c) Representación de la propuesta para algoritmos que admiten dos matrices a su entrada y devuelven una nueva a su salida; (d) Representación de la propuesta para algoritmos que admiten dos matrices a su entrada y devuelven un valor simple a su salida.

El grupo de algoritmos que manejan una o dos estructuras y que devuelven un valor simple obtendrían una mejora en la representación de sus ejecuciones a través de SRec si la vista cronológica añadiera en un lateral la información sobre el valor de retorno de manera asociada a cada solución parcial obtenida al finalizar cada llamada recursiva del algoritmo. De esta forma, la vista conseguiría ofrecer la información sobre la ejecución en un estado totalmente completo, como refleja la Figura 9.

Todas estas representaciones propuestas son, en mayor o menor grado, variaciones o ampliaciones de las visualizaciones que actualmente ya produce SRec. Suponen aún un campo no explorado totalmente y se deja su continuación como trabajo futuro para una catalogación más exhaustiva, en esta presentación se pretende establecer las bases de las propuestas, contextualizadas en una búsqueda bibliográfica que ha permitido identificar una serie de carencias y limitaciones que se pretenden superar.

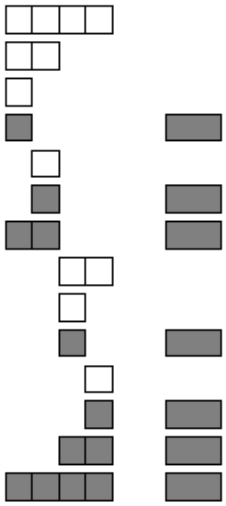

Figura 9 – Las posiciones blancas representan elementos de entrada, mientras que las posiciones de color gris oscuro representan valores de salida; Representación de la propuesta para la vista cronológica de SRec que permite complementar la información proporcionada sobre la estructura con los resultados que se van obteniendo de ella.

4. Conclusiones

Se ha repasado una quincena de libros dedicados a la algoritmia para evaluar la cantidad y variedad de las representaciones gráficas empleadas a la hora de ilustrar los problemas de la técnica "divide y vencerás". Se ha catalogado el repertorio de problemas encontrados en varias categorías para facilitar la extrapolación de conclusiones acerca de la disponibilidad, calidad y variedad de las representaciones en función de una serie de características comunes entre tales problemas.

Esto ha permitido además identificar para qué tipos de problemas SRec ofrece visualizaciones aptas para su utilización y para qué categorías SRec no es capaz

de proporcionar herramientas que ayuden en la comprensión o el análisis. Una de éstas es la de algoritmos geométricos, en donde SRec ofrece visualizaciones genéricas que no ayudan de manera significativa en las tareas del aprendizaje de los alumnos.

Es por tanto un trabajo futuro indagar en qué posibilidades existen para que SRec incorpore visualizaciones adaptadas a este tipo de problemas, si bien éstas no entrarían dentro de la clasificación de representaciones genéricas al ser dependientes del dominio, rompiendo así la estrategia seguida hasta ahora en el desarrollo de esta herramienta educativa.

No obstante, se ha comprobado que para múltiples tipos de problemas SRec sí ofrece soluciones adecuadas, aptas para su utilización con distintos fines (comprensión, análisis, presentación, elaboración de documentación mediante su capacidad de exportación, etc.).

Otro trabajo futuro es la propuesta de más tipos de representaciones para más problemas, no abordadas en esta comunicación. Es éste un reto importante que permitirá ampliar el catálogo de problemas representables de cara a su utilización en clases por parte de profesores y estudiantes.

5. Agradecimientos

Este trabajo ha sido financiado por el proyecto TIN2008-04103 del Ministerio de Ciencia e Innovación del Gobierno de España.

Referências Bibliográficas

Velázquez-Iturbide, J.Á., Pérez-Carrasco, A. (2008). *SRec: an animation system of recursion for algorithm courses*. Proceedings of the 13th annual conference on Innovation and technology in computer science education (ITICSE 08), SIGCSE Bull. Ed. ACM (40)3, 225-229.

Velázquez-Iturbide, J.Á., Pérez-Carrasco, A. (2009). *A Design of Automatic Visualizations for Divide-and-Conquer Algorithms. Electronic* Notes in Theoretical Computer Science, 224 (January 2009), 159-167.

Stern, L., Naish, L. (2002). *Visual representations for recursive algorithms*. Proceedings of the 33rd SIGCSE technical symposium on Computer science education, ACM SIGCSE Bulletin, 34(1), 196-200.

Di Batista, P., Eades, G., Tamassia, T., Toillis, I. (1999). *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall (Cap. 3).

Johnsonbaugh, R., Schaefer, M. (2004): *Algorithms (internacional edition)*. Pearson Education (Cap. 5).

Sahni, S. (2005). *Data structures, Algorithms and Applications in Java*. Silicon Press (Cap. 19).

Cormen, T.H., Leiserson, C.E., Rivest, R.L. (2005). *Introduction to algorithms*. The MIT Press.

Alsuwaiyel, M. H. (1999). *Algorithms, design techniques and analysis*. World Scientific (Cap. 6).

Brassad, G., Bratley, P. (1997). *Fundamentos de algoritmia*. Prentice Hall (Cap. 7)

Kleinberg, J., Tardos, É.(2006). *Algorithm design*. Pearson Addison-Wesley (Cap. 5).

De Giusti, A. (2001). *Algoritmos, datos y programas con aplicaciones en Pascal, Delphi y Visual Da Vinci*. Prentice Hall (Cap. 7).

Brassad, G., Bratley, P. (1996). *Algorítmica, concepción y análisis*. Masson (Cap. 4).

Lee, R.C.T., Tseng, S.S., Chang R.C., Tsai, Y.T. (2007). *Introducción al diseño y análisis de algoritmos, un enfoque estratégico*. Mc Graw Hill (Cap. 4).

Martí N. (2004). *Estructuras de datos y métodos algorítmicos ejercicios resueltos*.

Pearson Prentice Hall (Cap. 11).

Gonzalo A. (1997). *Esquemas algorítmicos: enfoque metodológico y problemas resueltos*. Universidad Nacional de Educación a Distancia (Cap. 3).

Allen, M. (1995). *Estructuras de datos y algoritmos.* Addison-Wesley Iberoamericana (Cap. 10.2).